

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-29788

(P2000-29788A)

(43) 公開日 平成12年1月28日 (2000.1.28)

(51) Int.Cl.⁷

識別記号

F I

キーワード (参考)

G 0 6 F 12/08

G 0 6 F 12/08

M 5 B 0 0 5

B

U

審査請求 有 請求項の数12 O L (全 13 頁)

(21) 出願番号 特願平10-199396

(22) 出願日 平成10年7月15日 (1998.7.15)

(71) 出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72) 発明者 撫原 恒平

東京都港区芝五丁目7番1号 日本電気株式会社内

(74) 代理人 100088812

弁理士 ▲柳▼川 信

Fターム (参考) 5B005 JJ11 KK12 MM01 NN01 NN12

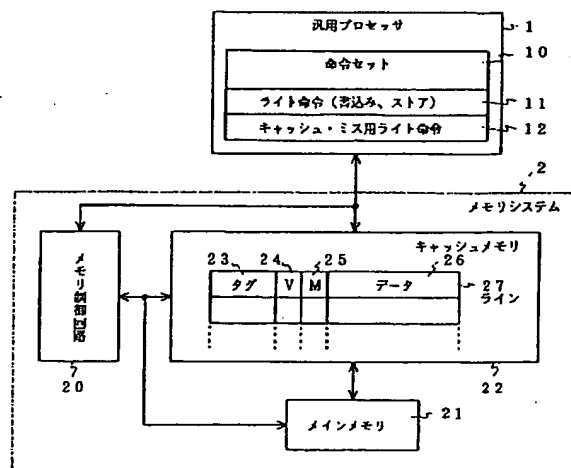
NN31 NN43 NN45 SS12

(54) 【発明の名称】 キャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体

(57) 【要約】

【課題】 連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化し、そのキャッシュ・ミス処理を含むソフトウェアの実行速度が改善可能なキャッシュメモリシステムを提供する。

【解決手段】 汎用プロセッサ1の命令セット10中に従来のキャッシュ制御手法に従う通常のライト命令 (書込み、ストア) 11と、本発明の一実施例によるキャッシュ制御手法に従うキャッシュ・ミス用ライト命令12とが夫々別の命令として用意されている。キャッシュ・ミスの処理を行う必要がある場合、キャッシュ・ミス用ライト命令12は書込みアドレスに相当するメインメモリ21のアドレスからデータを読み込むことなく、キャッシュ・ライン27を書込みデータで更新し、そのライン27のVビット24及びMビット25の双方を“1”にセットする。



【特許請求の範囲】

【請求項1】 メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムであって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択する選択手段と、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定する判定手段と、前記判定手段が当該データを無効と判定した時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読み込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込む書込み手段とを有することを特徴とするキャッシュメモリシステム。

【請求項2】 前記選択手段は、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライト対象のデータを格納するアドレスを選択するよう構成したことを特徴とする請求項1記載のキャッシュメモリシステム。

【請求項3】 前記判定手段は、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効を示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しいことを示す場合とのうちのいずれかの場合に前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定するよう構成したことを特徴とする請求項1または請求項2記載のキャッシュメモリシステム。

【請求項4】 前記書込み手段が前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込んだ後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットしかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットする手段を含むことを特徴とする請求項3記載のキャッシュメモリシステム。

【請求項5】 メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムのキャッシュ制御方法であって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択するステップと、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定するステ

ップと、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読み込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込むステップとを有することを特徴とするキャッシュ制御方法。

【請求項6】 前記キャッシュメモリのアドレスを選択するステップは、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライト対象のデータを格納するアドレスを選択するようにしたことを特徴とする請求項5記載のキャッシュ制御方法。

【請求項7】 前記前記キャッシュメモリのデータが有効か無効かを判定するステップは、前記選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効を示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しいことを示す場合とのうちのいずれかの場合に前記選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定するようにしたことを特徴とする請求項5または請求項6記載のキャッシュ制御方法。

【請求項8】 前記ライトキャッシュ時のデータが前記選択手段で選択されたアドレスに書込まれた後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットしかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットするステップを含むことを特徴とする請求項7記載のキャッシュ制御方法。

【請求項9】 コンピュータに、メインメモリに格納されたデータの一部を保持するキャッシュメモリを制御させるためのキャッシュ制御プログラムを記録した記録媒体であって、前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択させ、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定させ、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読み込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込ませることを特徴とするキャッシュ制御プログラムを記録した記録媒体。

【請求項10】 前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリのアドレスを選択させる際に、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッ

メモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライト対象のデータを格納するアドレスを選択させることを特徴とする請求項9記載のキャッシュ制御プログラムを記録した記録媒体。

【請求項11】 前記キャッシュ制御プログラムは前記コンピュータに、前記前記キャッシュメモリのデータが有効か無効かを判定させる際に、前記選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しくないことを示す場合とのうちのいずれかの場合に前記選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定させることを特徴とする請求項9または請求項10記載のキャッシュ制御プログラムを記録した記録媒体。

【請求項12】 前記キャッシュ制御プログラムは前記コンピュータに、前記ライトキャッシュ時のデータが前記選択手段で選択されたアドレスに書込まれた後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットさせかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットさせることを特徴とする請求項11記載のキャッシュ制御プログラムを記録した記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明はキャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体に関し、特にマイクロプロセッサ等のキャッシュメモリシステムにおけるライト動作を高速化するためのキャッシュ制御方法に関する。

【0002】

【従来の技術】現在、マルチメディア処理はテレビゲーム機やインターネット端末といった安価な民生用家電製品から、カーナビゲーションシステムやパーソナルコンピュータといった比較的高性能な製品まで、様々な情報機器で必要とされている。これらの機器で用いられるマルチメディア処理には音声、オーディオ、ビデオ信号各々の圧縮や伸張、及び2次元や3次元のグラフィックス処理等が含まれる。これらのマルチメディア処理では膨大なデータに対し、多くの演算を行う必要がある。

【0003】従来、上記のマルチメディア処理においては、機器全体の制御やユーザインタフェースを行う汎用プロセッサに加え、特定のマルチメディア処理を専用に行うLSI（大規模集積回路）を機器へ組込むことで、処理能力の要求を満たしてきている。

【0004】しかしながら、最近では、汎用プロセッサ

の性能向上によって、専用LSIを組込まなくても、汎用プロセッサ上のソフトウェアによって十分な速度でマルチメディア処理が行えるようになりつつある。

【0005】専用LSIを汎用プロセッサ上のマルチメディア処理ソフトウェアで代替すると、専用LSIという付加ハードウェアが不要になるため、機器を安価に構成することができ、その機能をソフトウェアの変更によって容易に実現することができる。例えば、ソフトウェアの変更によって、オーディオの伸張とビデオ信号の伸張とを切替えることができる。

【0006】上記のように、ソフトウェアによるマルチメディア処理は利点が多いため、現在の汎用プロセッサでは処理能力が不足する応用でもソフトウェア処理でカバーしようと、汎用プロセッサにマルチメディア処理専用の演算命令を導入し、マルチメディア処理性能を向上させることも一般的である。

【0007】上記のようなマルチメディア処理命令については、“VIS Speeds New Media Processing,” (Marc Tremblayほか, IEEE Micro, pp. 10~20, Aug. 1996)、“Intel’s MMX Speeds Multimedia” (Linley Gwennap, Microprocessor Report, Vol. 10, No. 3, pp. 1, 6~10, Mar. 5, 1996)、“NEC V830R/AV Handles Real-Time MPEG-2” (Jim Turley, Microprocessor Report, Vol. 11, No. 12, pp. 5, Sep. 15, 1997)等に記載されている。

【0008】上記の論文に記載されたマルチメディア処理専用の命令セットはレジスタを分割して複数のデータを格納し、1命令で並列に演算を行うことで、データ並列性の高い処理における処理能力を向上させている。例えば、64ビット長のレジスタに16ビット長のデータ4個を格納し、1命令でこれらのデータに対して4並列演算を行える。このようなデータ並列を利用する演算命令を以下、SIMD (Single-Instruction Multiple-Data) 命令と呼ぶ。

【0009】汎用プロセッサで効率良くマルチメディア処理を行うには、SIMD命令を導入して演算能力を強化すると同時に、メモリシステムを強化し、演算能力に見合ったレートでメモリシステム上のデータが演算器へ供給されるように、また演算後のデータがメモリシステムに格納されるようにすることが重要である。

【0010】図4～図6を参照して汎用プロセッサ5を用いたシステムの典型的なメモリシステム2の構成及びデータアクセス時の動作について説明する。図4に示すように、汎用プロセッサ5に接続されるメモリシステム2は大容量なメインメモリ21と、高速なキャッシュメモリ22とを組合わせて構成され、メインメモリ21及

びキャッシュメモリ22を制御するメモリ制御回路20を備えている。

【0011】メインメモリ21に格納されたデータのうち、頻繁にアクセスされる一部のデータのコピーがキャッシュメモリ22に格納され、汎用プロセッサ5がデータアクセスを行う時に、キャッシュメモリ22上に目的のデータが存在すれば、低速なメインメモリ21ではなく、高速なキャッシュメモリ22をアクセスするよう制御し、プログラムの実行速度を改善する。

【0012】キャッシュメモリ22は通常、数百〜数千個のライン27に分割され、ライン27毎に制御情報を付加し、状態を管理する。1個のライン27が保持するデータのサイズは通常、16〜64バイトである。制御情報としては、ライン27毎に、保持しているデータ26のメインメモリ21上のアドレスの上位ビットを示すタグ23と、データ26が有効であることを示すバリッド(V)ビット24と、データ26が更新されたことを示すモディファイ(M)ビット25とが付加される。

【0013】Vビット24はそのライン27のデータ26が有効か無効かを示す。以下、Vビット24が“1”の時にはそのライン27のデータ26を有効とし、“0”の時にはデータ26を無効とする。

【0014】Mビット25はそのライン27のデータ26が更新されており、キャッシュメモリ22上のデータ26がメインメモリ21上のデータより新しいか、更新されておらず、キャッシュメモリ22上のデータ26とメインメモリ21上のデータとが等しいかを示す。以下、Mビットが“1”の時にはキャッシュメモリ22上のデータ26がメインメモリ21上のデータより新しいとし、“0”の時にはキャッシュメモリ22上のデータ26とメインメモリ21上のデータとが等しいものとする。

【0015】図4に示すように、汎用プロセッサ5とキャッシュメモリ22とは論理的に独立した機能を持っているが、物理的には同一のLSIチップ上に集積されている場合、別チップだが同一のマルチチップモジュールやカートリッジ内に組込まれている場合等がある。

【0016】また、図4に示す例では汎用プロセッサ5とメインメモリ21との間に存在するキャッシュメモリ22は1段だが、より高性能なメモリシステム2を構成するためにはキャッシュメモリ22を多段にすることもあ

る。

【0017】データリード時のキャッシュメモリ22の制御について、図5を参照しながら説明する。汎用プロセッサ5からキャッシュメモリ22にアドレスArのデータのリード要求があったとする。メモリ制御回路20はアドレスArの上位ビットと同じタグ23の値を持つライン27を検索する(図5ステップS11)。

【0018】アドレスArの上位ビットと同じタグ23の値を持つライン27が存在し、このラインをラインi

とすると、メモリ制御回路20はラインiのVビット24を調べ(図5ステップS12)、Vビット24が“1”ならば、ラインiが目的のデータを含んでいるため(キャッシュ・ヒット)、ラインiのデータ26のうち必要とされる部分を汎用プロセッサ5に返す(図5ステップS13)。

【0019】キャッシュメモリ22中に、アドレスArの上位ビットと同じタグ23の値を持つラインが存在しない時、または存在してもVビット24が“0”の時、キャッシュメモリ22中に目的のデータが存在しない(キャッシュ・ミス)ので、メモリ制御回路20は適当なライン(ラインjとする)を選択し(図5ステップS14)、メインメモリ21上のアドレスArのデータをラインjにコピーする(図5ステップS18)。

【0020】但し、メモリ制御回路20は選択したラインjのVビット24及びMビット25を調べ(図5ステップS15、S16)、それらがともに“1”の時にはラインjがメインメモリ21よりも新しいデータを保持しているので、ラインjのデータをメインメモリ21の対応するアドレスへ書き戻した後(図5ステップS17)、アドレスArのデータをラインjにコピーする(図5ステップS18)。

【0021】その後、メモリ制御回路20はVビット24を“1”に、Mビット25を“0”に夫々設定し(図5ステップS19)、ラインjがメインメモリ21のデータの有効なコピーを保持していることを示し、ラインjのデータのうち必要な部分を汎用プロセッサ5へ返す(図5ステップS20)。このとき、ライン27の値jは通常、アドレスArの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【0022】次に、データライト時のキャッシュメモリ22の制御について、図6を参照しながら説明する。ここでは、特にライト・バック方式と呼ばれるキャッシュ制御方式について説明する。尚、このデータライトは汎用プロセッサ5内の命令セット50に含まれるライト命令(書込み、ストア)51によって実行される。

【0023】汎用プロセッサ5からキャッシュメモリ22にアドレスAwのデータのライト要求があったとする。メモリ制御回路20はアドレスAwの上位ビットと同じタグ23の値を持つライン27を検索する(図6ステップS31)。

【0024】アドレスAwの上位ビットと同じタグ23の値を持つライン27がキャッシュメモリ22上に存在し、そのラインをラインiとすると、メモリ制御回路20はラインiのVビット24を調べ(図6ステップS32)、Vビット24が“1”ならば、ラインiが目的のデータを含んでいるため(キャッシュ・ヒット)、ラインi上のデータを書換える(図6ステップS33)。さらに、メモリ制御回路20はラインiのMビットを

“1”にし(図6ステップS34)、メインメモリ21上のデータよりキャッシュ・ラインi上のデータが新しいことを示す。

【0025】キャッシュメモリ22中に、アドレスAwの上位ビットと同じタグ値を持つラインが存在しない時、または存在してもVビット24が“0”の時、キャッシュメモリ22中に目的のデータは存在しない(キャッシュ・ミス)ので、メモリ制御回路20は適当なライン(ラインjとする)を選択し(図6ステップS35)、メインメモリ21上のアドレスAwのデータをラ

インjにコピーする(図6ステップS39)。
【0026】但し、メモリ制御回路20は選択したラインjのVビット24及びMビット25を調べ(図6ステップS36、S37)、それらがともに“1”の時にはラインjがメインメモリ21よりも新しいデータを保持しているので、ラインjのデータをメインメモリ21の対応するアドレスへ書戻した後(図6ステップS38)、アドレスAwのデータをラインjにコピーする(図6ステップS39)。

【0027】その後、メモリ制御回路20はラインjのデータを更新し(図6ステップS40)、ラインjのVビット24及びMビット25をともに“1”にし(図6ステップS41)、キャッシュメモリ22のラインj上に有効なデータがあり、メインメモリ21上のデータよりキャッシュメモリ22のラインj上のデータが新しいことを示す。リードキャッシュ・ミスの時と同様に、ライン27の値jは通常、アドレスAwの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【0028】ライトキャッシュ・ミスの時、アドレスAwにある値をいったんキャッシュメモリ22に読込むのは、一度に書換えられるデータがキャッシュ・ラインjの一部のみだからである。キャッシュの1ラインは通常、16～64バイトのデータを保持するが、1個のデータ書き込み命令は通常、1～8バイトのデータを書込む。キャッシュ・ラインjのデータのうち、データ書き込みによって更新されなかったデータは、メインメモリ21上で対応するデータの値と一致していなくては整合性が保たれない。

【0029】

【発明が解決しようとする課題】一般的に、マルチメディア処理では大量の演算結果を連続したアドレスへ書込むことが多い。例えば、ビデオ画像の圧縮規格であるITU-T勧告H. 261、H. 262、H. 263、MPEG(Moving Picture Experts Group)-1ビデオ(ISO/IEC11172、“Information Technology-Coding of Moving Pictures and Associated Audio for Digital Storage Media u

p tol. 5Mbit/s”)、MPEG-2ビデオ(ISO/IEC13818-2、“Information Technology-Generic Coding of of Moving Pictures and Associated Audio”)に従って圧縮されたビットストリームを伸張し、ビデオ画像を再生する場合、再生画像はメインメモリ上に確保したフレームバッファに書込まれる。

【0030】例えば、MPEG-2 Main Levelで一般的に用いられる解像度である720×480画素の画像を伸張した場合、1ピクチャあたり輝度信号に345、600バイトの連続した領域を、色差信号2枚に各86、400バイトの連続した領域をメインメモリ上に確保する。

【0031】すなわち、1ピクチャあたり506Kバイトのメモリ量が必要である。MPEG-2で用いられる双方向動き予測を実現するには最低でも3枚のピクチャを保持する必要があるため、MPEG-2ビデオ伸張ソフトウェアの作業領域は最低でも1.5Mバイト必要になる。キャッシュメモリの容量は通常数十～数百Kバイトであり、これだけの量のデータをすべて保持することはできない。したがって、フレームバッファアクセスではデータキャッシュ・ミスが頻発し、高速なMPEG-2ビデオ伸張ソフトウェア作成のための障害となっている。

【0032】さらに、1秒間に30枚の画像をリアルタイム再生するには、毎秒14.8Mバイトのデータをメインメモリに書込む必要がある。この書込みがデータキャッシュ・ミスを引き起こしていると考えると、キャッシュメモリとメインメモリとの整合性を保つために、伸張データの書込みの前に、メインメモリからキャッシュメモリへ同じ量のデータを読込んでいるため、この2倍、毎秒29.6Mバイトのデータアクセスがメインメモリに対して発生していることになる。この量は、例えば、32bit幅66MHz動作のバスのバンド幅の11.5%にも達し、その処理性能上無視することができない。

【0033】図7に連続したアドレスへデータをライトした時のキャッシュメモリとメインメモリとの間のデータのやり取りと、キャッシュメモリ及びメインメモリ上のデータの変化とを示す。

【0034】典型的な例として、キャッシュの1ラインのデータサイズを16バイト、ライトするデータのサイズを4バイトと仮定する。ここで、メインメモリのアドレスAwから4個の4バイト・データN0、N1、N2、N3を1個ずつライトした場合を考える。

【0035】さらに、キャッシュメモリ上にはアドレスAwのデータが最初存在しないものとし、アドレスAwのデータを保持するキャッシュ・ラインiは不定値U0、U1、U2、U3を格納しているものとする。

【0036】最初に、アドレスAwに値N0をライトした場合の動作を考える。アドレスAwへの値N0のライト要求が発生した時〔図7の(11)参照〕、アドレスAwのデータはキャッシュメモリ上に存在しないためにキャッシュ・ミスが発生し、メインメモリ上のアドレスAwから1ライン16バイト分のデータM0、M1、M2、M3〔図7の47参照〕がキャッシュ・ラインiに読込まれる〔図7の(17)及び42参照〕。その後、ラインiのデータのうちアドレスAwからの4バイト分のデータに相当するM0がN0によって上書きされる〔図7の(12)及び43参照〕。

【0037】次に、隣接するアドレスAw+4に4バイトのデータN1がライトされる〔図7の(13)参照〕。アドレスAwから1ライン16バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインi上のアドレスAw+4に相当するデータをN1に更新する〔図7の44参照〕。

【0038】三番目に、隣接するアドレスAw+8に、4バイトのデータN2がライトされる〔図7の(14)参照〕。アドレスAwから1ライン16バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインi上のアドレスAw+8に相当するデータをN2に更新する〔図7の45参照〕。

【0039】最後に、隣接するアドレスAw+12に、4バイトのデータN3がライトされる〔図7の(15)参照〕。アドレスAwから1ライン16バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインi上のアドレスAw+12に相当するデータをN3に更新する〔図7の46参照〕。

【0040】以上、N0、N1、N2、N3のライトによって、キャッシュ・ラインi上のデータはすべて書換えられており、メインメモリ上のアドレスAwから1ライン16バイト分のデータ47よりも新しい。したがって、ラインiに別のアドレスのデータを格納する時には、まずラインiのデータをメインメモリに書戻す必要がある〔図7の(16)及び48参照〕。

【0041】そこで、本発明の目的は上記の問題点を解消し、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができるキャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体を提供することにある。

【0042】

〔課題を解決するための手段〕本発明によるキャッシュメモリシステムは、メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムであって、前記キャ

ッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択する選択手段と、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定する判定手段と、前記判定手段が当該データを無効と判定した時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込む書込み手段とを備えている。

【0043】本発明によるキャッシュ制御方法は、メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムのキャッシュ制御方法であって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択するステップと、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定するステップと、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込むステップとを備えている。

【0044】本発明によるキャッシュ制御プログラムを記録した記録媒体は、コンピュータに、メインメモリに格納されたデータの一部を保持するキャッシュメモリを制御させるためのキャッシュ制御プログラムを記録した記録媒体であって、前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択させ、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定させ、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込ませている。

【0045】すなわち、本発明のキャッシュメモリシステムは、データと、保持しているデータのメインメモリ上のアドレスの上位ビットを示すタグと、データが有効であることを示すバリッド(V)ビットと、データが更新されたことを示すモディファイ(M)ビットとを含む複数のキャッシュ・ラインから構成されている。

【0046】本発明のキャッシュメモリシステムは、キャッシュメモリへの書込み時、キャッシュ中に書込みアドレスの上位ビットと一致するタグを持ち、かつVビットが“1”であるラインが存在せず、キャッシュ・ミスの処理を行う必要がある時、書込みアドレスに相当するメインメモリのアドレスからデータを読込むことなく、キャッシュ・ラインを書込みデータで更新し、Vビット

10

20

30

40

50

及びMビットの双方を“1”にセットしている。

【0047】このため、メインメモリのアドレスからデータを読み込む時間を削減できる。よって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することが可能となり、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することが可能となる。

【0048】

【発明の実施の形態】次に、本発明の一実施例について図面を参照して説明する。図1は本発明の一実施例によるキャッシュメモリシステムの構成を示すブロック図である。図において、本発明の一実施例によるキャッシュメモリシステムは汎用プロセッサ1と、汎用プロセッサ1に接続されるメモリ・システム2とから構成されている。

【0049】汎用プロセッサ1はライト命令（書き込み、ストア）11とキャッシュミス用ライト命令12を含む命令セット10を持ち、ソフトウェア実行時、メモリシステム2に格納されたデータをリード・ライトする。尚、命令セット10はROM（リードオンリメモリ）やIC（集積回路）メモリ等に格納可能である。

【0050】メモリシステム2は大容量なメインメモリ21と、高速なキャッシュメモリ22とを組合わせて構成され、メインメモリ21及びキャッシュメモリ22を制御するメモリ制御回路20を備えている。

【0051】キャッシュメモリ22は数百～数千個のライン27に分割され、ライン27毎に制御情報を付加し、状態を管理する。1個のライン27が保持するデータのサイズは16～64バイトである。

【0052】制御情報としては1ライン27毎に、保持しているデータ26のメインメモリ21上のアドレスの上位ビットを示すタグ23と、データ26が有効であることを示すバリッド（V）ビット24と、データ26が更新されたことを示すモディファイ（M）ビット25とが付加される。

【0053】Vビット24はそのライン27のデータ26が有効か無効かを示す。以下、Vビット24が“1”の時にはそのライン27のデータ26を有効とし、“0”の時にはデータ26を無効とする。

【0054】Mビット25はそのライン27のデータ26が更新されており、キャッシュメモリ22上のデータ26がメインメモリ21上のデータより新しいか、更新されておらず、キャッシュメモリ22上のデータ26とメインメモリ21上のデータとが等しいかを示す。以下、Mビットが“1”の時にはキャッシュメモリ22上のデータ26がメインメモリ21上のデータより新しいとし、“0”の時にはキャッシュメモリ22上のデータ26とメインメモリ21上のデータとが等しいものとする。

【0055】本発明の一実施例においても、データリー

ド時の動作は図5に示す従来の技術と同様である。すなわち、汎用プロセッサ1からキャッシュメモリ22にアドレスArのデータのリード要求があったとする。メモリ制御回路20はアドレスArの上位ビットと同じタグ23の値を持つライン27を検索する（図5ステップS11）。

【0056】アドレスArの上位ビットと同じタグ23の値を持つライン27が存在し、このラインをラインiとすると、メモリ制御回路20はラインiのVビット24を調べ（図5ステップS12）、Vビット24が“1”ならば、ラインiが目的のデータを含んでいるため（キャッシュ・ヒット）、ラインiのデータ26のうち必要とされる部分を汎用プロセッサ1に返す（図5ステップS13）。

【0057】キャッシュメモリ22中に、アドレスArの上位ビットと同じタグ23の値を持つラインが存在しない時、または存在してもVビット24が“0”の時、キャッシュメモリ22中に目的のデータが存在しない（キャッシュ・ミス）ので、メモリ制御回路20は適当なライン（ラインjとする）を選択し（図5ステップS14）、メインメモリ21上のアドレスArのデータをラインjにコピーする（図5ステップS18）。

【0058】但し、メモリ制御回路20は選択したラインjのVビット24及びMビット25を調べ（図5ステップS15、S16）、それらがともに“1”の時にはラインjがメインメモリ21よりも新しいデータを保持しているので、ラインjのデータをメインメモリ21の対応するアドレスへ書き戻した後（図5ステップS17）、アドレスArのデータをラインjにコピーする（図5ステップS18）。

【0059】その後、メモリ制御回路20はVビット24を“1”に、Mビット25を“0”に夫々設定し（図5ステップS19）、ラインjがメインメモリ21のデータの有効なコピーを保持していることを示し、ラインjのデータのうち必要な部分を汎用プロセッサ1へ返す（図5ステップS20）。このとき、ライン27の値jは通常、アドレスArの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【0060】図2は本発明の一実施例によるデータライト時の動作を示すフローチャートである。これら図1及び図2を参照して本発明の一実施例によるデータライト時の動作について説明する。

【0061】汎用プロセッサ1からキャッシュメモリ22にアドレスAwのデータのライト要求があったとする。メモリ制御回路20はアドレスAwの上位ビットと同じタグ23の値を持つライン27を検索する（図2ステップS1）。

【0062】アドレスAwの上位ビットと同じタグ23の値を持つライン27が存在し、このラインをラインi

とすると、メモリ制御回路20はラインiのVビット24を調べ(図2ステップS2)、Vビット24が“1”ならば、ラインiが目的のデータを含んでいるため(キャッシュ・ヒット)、ラインi上のデータを書換える(図2ステップS3)。さらに、メモリ制御回路20はラインiのMビットを“1”にし(図2ステップS4)、メインメモリ21上のデータよりキャッシュ・ラインi上のデータが新しいことを示す。

【0063】キャッシュメモリ22中に、アドレスAwの上位ビットと同じタグ23の値を持つラインが存在しない時、または存在してもVビット24が“0”の時、キャッシュメモリ22中に目的のデータは存在しない(キャッシュ・ミス)ので、メモリ制御回路20は適当なライン(ラインjとする)を選択し(図2ステップS5)、その後、ラインjにデータをライトする(図2ステップS9)。

【0064】但し、メモリ制御回路20は選択したラインjのVビット24及びMビット25を調べ(図2ステップS6、S7)、それらがともに“1”の時にはラインjがメインメモリ21よりも新しいデータを保持しているので、ラインjのデータをメインメモリ21の対応するアドレスへ書き戻した後(図2ステップS8)、ラインjにデータをライトする(図2ステップS9)。

【0065】その後、メモリ制御回路20はラインjのVビット24及びMビット25をとともに“1”にし(図2ステップS10)、メインメモリ21上のデータよりキャッシュ・ラインi上のデータが新しいことを示す。リードキャッシュ・ミス時と同様に、ライン27の値jは通常、アドレスAwの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【0066】図3は本発明の一実施例によるデータライト時の動作を示す図である。この図3を参照して本発明の一実施例によってキャッシュ・ミス時の処理がどのように高速化されるのかを説明する。

【0067】図3は図7に示す動作と同様に、連続したアドレスへデータをライトした時のキャッシュメモリ制御手法で制御されるキャッシュメモリ22とメインメモリ21との間のデータのやり取りと、キャッシュメモリ22及びメインメモリ21上のデータの変化とを示す。

【0068】この場合、図7と同様に、キャッシュの1ラインのデータサイズを16バイト、ライトするデータのサイズを4バイトと仮定し、メインメモリ21のアドレスAwから4個の4バイトのデータN0、N1、N2、N3を1個ずつライトした場合を考える。さらに、これも図7と同様に、開始時にキャッシュメモリ22上にはアドレスAwのデータが存在しないものとし、アドレスAwのデータを保持する予定のキャッシュ・ラインiは不定値U0、U1、U2、U3を格納しているものとする[図3の31参照]。

【0069】最初に、アドレスAwに値N0をライトする要求が発生した場合[図3の(1)参照]の動作を考える。このとき、アドレスAwのデータはキャッシュ上に存在しないため、キャッシュ・ミスが発生するが、本発明の一実施例では従来例とは異なり、メインメモリ21上のアドレスAwから1ライン分のデータM0、M1、M2、M3をキャッシュ・ラインiに読み込むことが行われず、ラインiのデータのうちアドレスAwからの4バイト分のデータに相当するU0がN0によって上書きされるのみである[図3の(2)及び32参照]。

【0070】ラインiのデータのうちアドレスAw+4、Aw+8、Aw+12に相当するデータは不定値U1、U2、U3であり、メインメモリ21の値M1、M2、M3を反映しない。しかしながら、キャッシュのヒット・ミス判定時にはラインiがメインメモリ21上のアドレスAwから1ライン16バイト分のデータを保持しているとみなされる。

【0071】次に、隣接するアドレスAw+4に4バイトのデータN1がライトされる[図3の(3)参照]。ヒット・ミス判定時にはアドレスAwから1ライン16バイト分のデータがキャッシュメモリ22上に存在すると扱われるため、このデータN1のライトはキャッシュにヒットし、ラインi上のアドレスAw+4に相当するデータがN1に更新される[図3の33参照]。

【0072】三番目に、隣接するアドレスAw+8に4バイトのデータN2がライトされる[図3の(4)参照]。ヒット・ミス判定時にはアドレスAwから1ライン16バイト分のデータがキャッシュメモリ22上に存在すると扱われるため、このデータN2のライトはキャッシュにヒットし、ラインi上のアドレスAw+8に相当するデータがN2に更新される[図3の34参照]。

【0073】最後に、隣接するアドレスAw+12に4バイトのデータN3がライトされる[図3の(5)参照]。ヒット・ミス判定時にはアドレスAwから1ライン16バイト分のデータがキャッシュメモリ22上に存在すると扱われるため、このライトはキャッシュにヒットし、ラインi上のアドレスAw+12に相当するデータがN3に更新される[図3の35参照]。

【0074】以上、N0、N1、N2、N3のライトによってキャッシュ・ラインi上のデータがすべて書き換えられており、メインメモリ21上のアドレスAwから1ライン16バイト分のデータ36よりも新しい。したがって、ラインiに別のアドレスのデータを格納する時には、まずラインiのデータをメインメモリ21に書き戻す必要がある[図3の(6)及び37参照]。

【0075】図3に示す本発明の一実施例による動作例と図7に示す従来の動作例とを比較すると、アドレスAwに値N3をライトした後は同一の結果が得られている。すなわち、キャッシュ・ラインiは値N0、N1、N2、N3を保持している[図3の35及び図7の4

6】。

【0076】しかしながら、従来例ではこの結果を得るためにメインメモリ21からキャッシュ・ラインiへのリード【図7の(17)】を行っているのに対し、本発明の一実施例ではこのリードを行っていない。したがって、本発明の一実施例では従来例よりも高速に結果を得ることができる。

【0077】図3において、キャッシュ・ミス時にメインメモリ21からキャッシュ・ラインへのデータリードが不要な理由を説明する。あるキャッシュ・ラインi上のデータが複数のライト動作ですべて更新される場合、最初のライトが起こしたキャッシュ・ミスでメインメモリ21からキャッシュ・ラインiへデータを転送しても、メインメモリ21から転送されたデータは複数のライト動作ですべて上書きされてしまうため、キャッシュ・ラインiが保持する最終結果はライトデータの値のみに依存し、メインメモリ21の内容には依存しない。したがって、メインメモリ21からキャッシュ・ラインへのデータリードは不要である。

【0078】一般的には、あるキャッシュ・ライン上のデータが複数のライト動作ですべて更新される保証はないため、メインメモリ21の内容と対応するキャッシュ・ラインのデータの内容との整合性を保つにはキャッシュ・ミス時にメインメモリ21の内容をキャッシュ・ラインにリードする必要がある。しかしながら、マルチメディアアプリケーション等ではしばしばキャッシュ・ライン上のデータが複数のライト動作ですべて更新されることが保証される。

【0079】例えば、圧縮されたビデオ画像をソフトウェアで伸張することを考える。伸張されたビデオ画像は数百キロバイト程度のフレームバッファに連続して書込まれるため、フレームバッファ領域に対応するキャッシュ・ラインはすべての内容が更新される。マルチメディアアプリケーション、特にビデオ画像を扱うアプリケーションでは大量のデータをフレームバッファに書込むことが多く、不要なリード動作を省略することによる高速化の効果も大きい。

【0080】図6に示す従来のキャッシュ制御手法に従う通常のライト命令（書込み、ストア）と、図2に示す本発明の一実施例によるキャッシュ制御手法に従うライト命令とは汎用プロセッサ1の命令セット10中に夫々別の命令として用意し、アプリケーションソフトウェアのプログラマが使い分ける。

【0081】なぜなら、あるキャッシュ・ライン上のデータが複数のライト動作ですべて更新されるかどうかを汎用プロセッサ1上の何らかのハードウェア機構で判断することは不可能だからである。また、高級言語コンパイラが同様の判断を下すことも困難だからである。

【0082】このように、汎用プロセッサ1の命令セット10中に従来のキャッシュ制御手法に従う通常のライ

ト命令（書込み、ストア）11と、図2に示す本発明の一実施例によるキャッシュ制御手法に従うキャッシュ・ミス用ライト命令12とを夫々別の命令として用意し、キャッシュ・ミスの処理を行う必要がある時にキャッシュ・ミス用ライト命令12によって書込みアドレスに相当するメインメモリ21のアドレスからデータを読み込むことなく、キャッシュ・ラインを書込みデータで更新し、そのラインのVビット及びMビットの双方を“1”にセットすることによって、あるキャッシュ・ライン上のデータをすべて更新する場合、ライト・ミス時のメインメモリ21からキャッシュ・ラインへのリードを省略してその処理を高速化することができる。

【0083】このため、メインメモリ21のアドレスからデータを読み込む時間を削減することができる。よって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができる。

【0084】

【発明の効果】以上説明したように本発明によれば、メインメモリと、メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムにおいて、キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納するキャッシュメモリのアドレスを選択し、その選択されたアドレスに対応するキャッシュメモリのデータが有効か無効かを判定し、当該データが無効と判定された時に当該データに対応するメインメモリのデータをキャッシュメモリに読み込むことなくライトキャッシュ時のデータを選択されたアドレスに書込むことによって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができるという効果がある。

【図面の簡単な説明】

【図1】本発明の一実施例によるキャッシュメモリシステムの構成を示すブロック図である。

【図2】本発明の一実施例によるデータライト時の動作を示すフローチャートである。

【図3】本発明の一実施例によるデータライト時の動作を示す図である。

【図4】従来例によるキャッシュメモリシステムの構成を示すブロック図である。

【図5】従来例によるデータリード時の動作を示すフローチャートである。

【図6】従来例によるデータライト時の動作を示すフローチャートである。

【図7】従来例によるデータライト時の動作を示す図である。

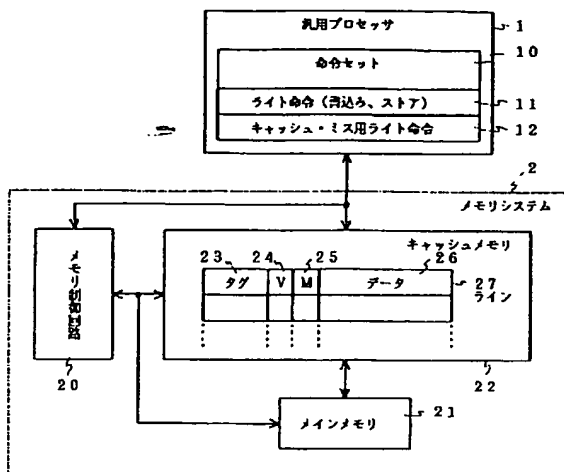
【符号の説明】

- 1 汎用プロセッサ
- 2 メモリシステム
- 10 命令セット
- 11 ライト命令（書込み、ストア）
- 12 キャッシュ・ミス用ライト命令
- 20 メモリ制御回路
- 21 メインメモリ

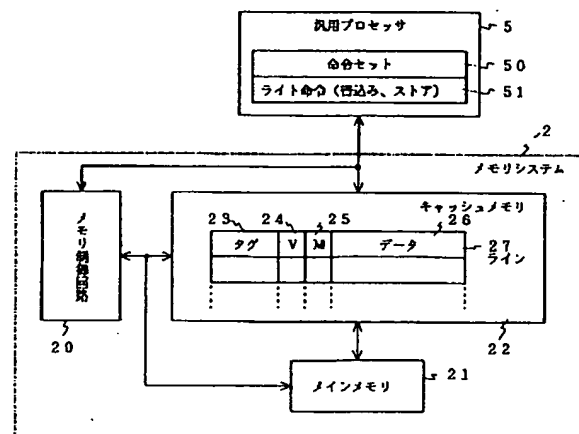
- * 22 キャッシュメモリ
- 23 タグ
- 24 バリッドビット
- 25 モディファイビット
- 26 データ
- 27 キャッシュ・ライン

*

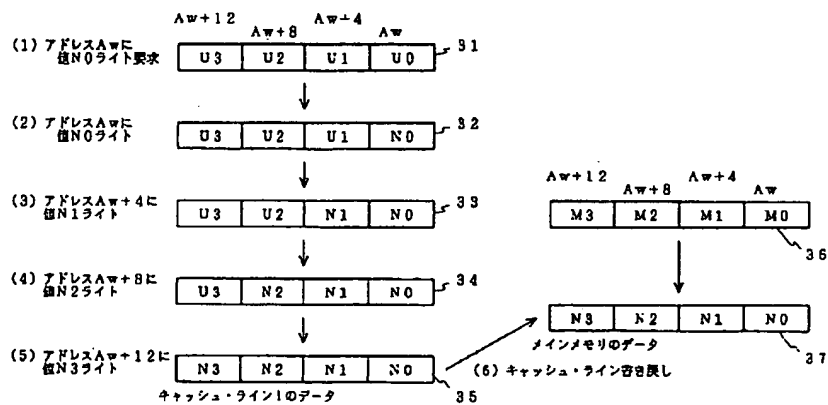
【図1】



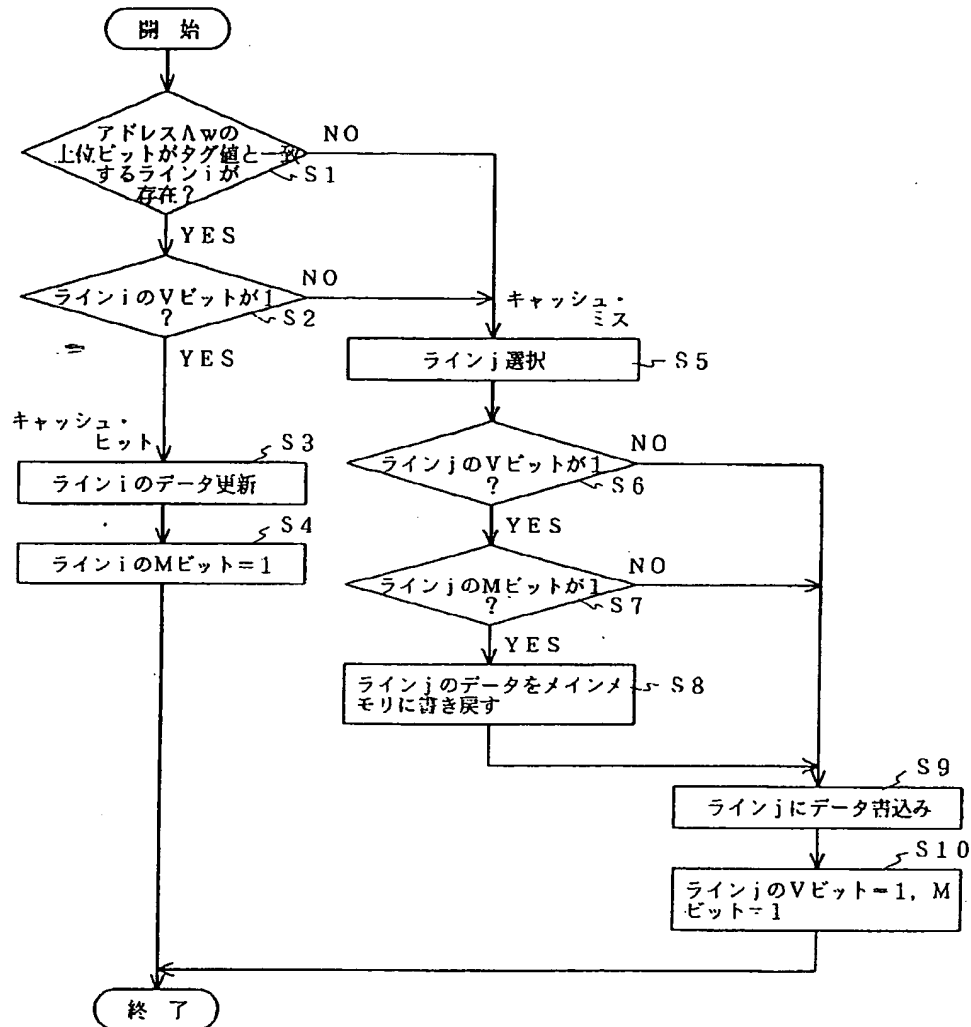
【図4】



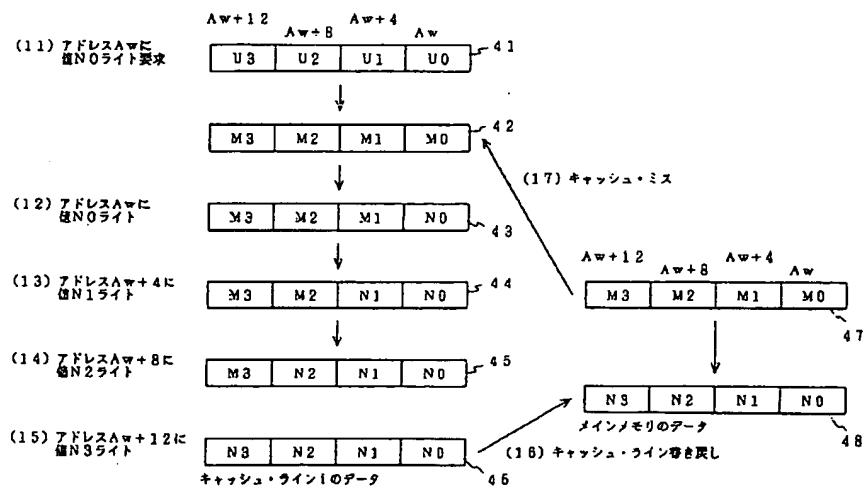
【図3】



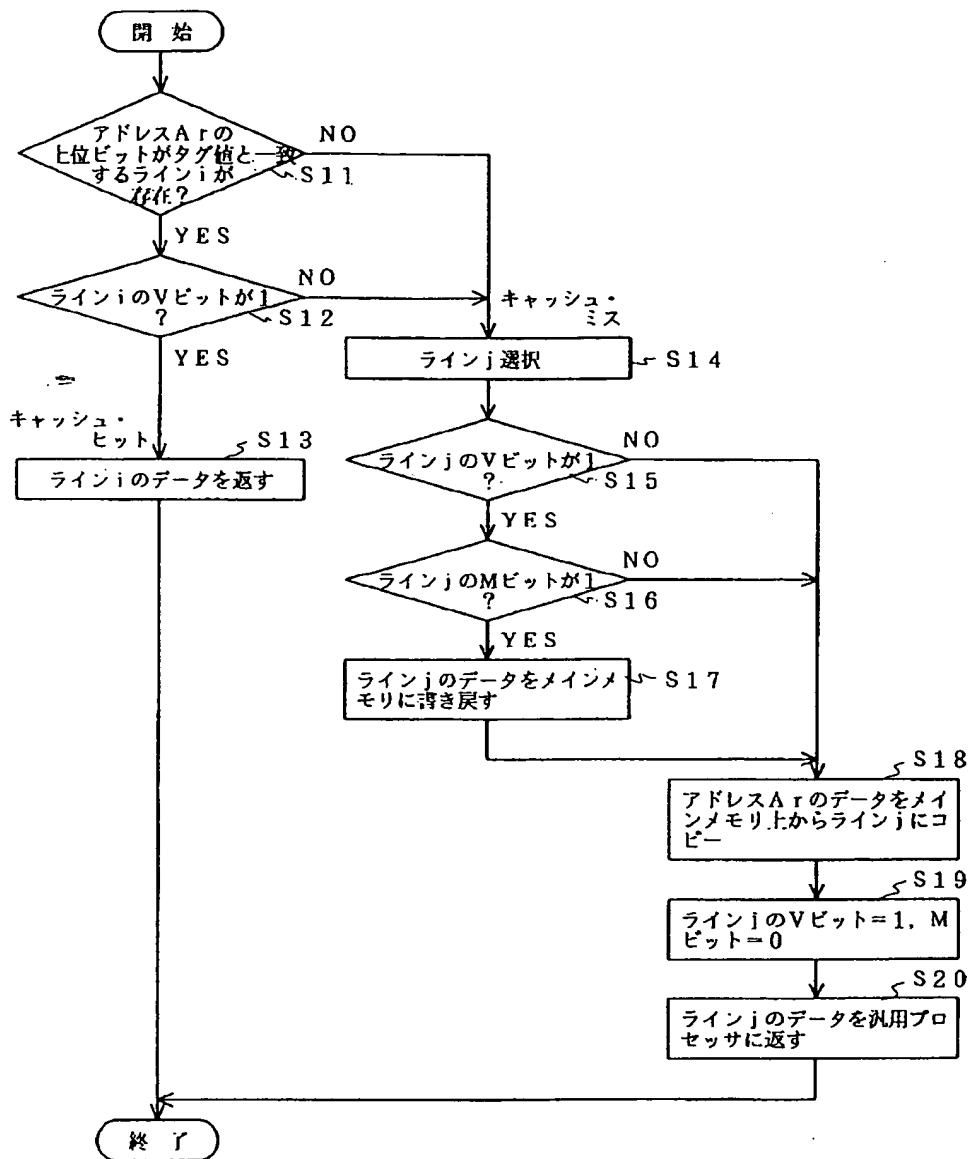
【図2】



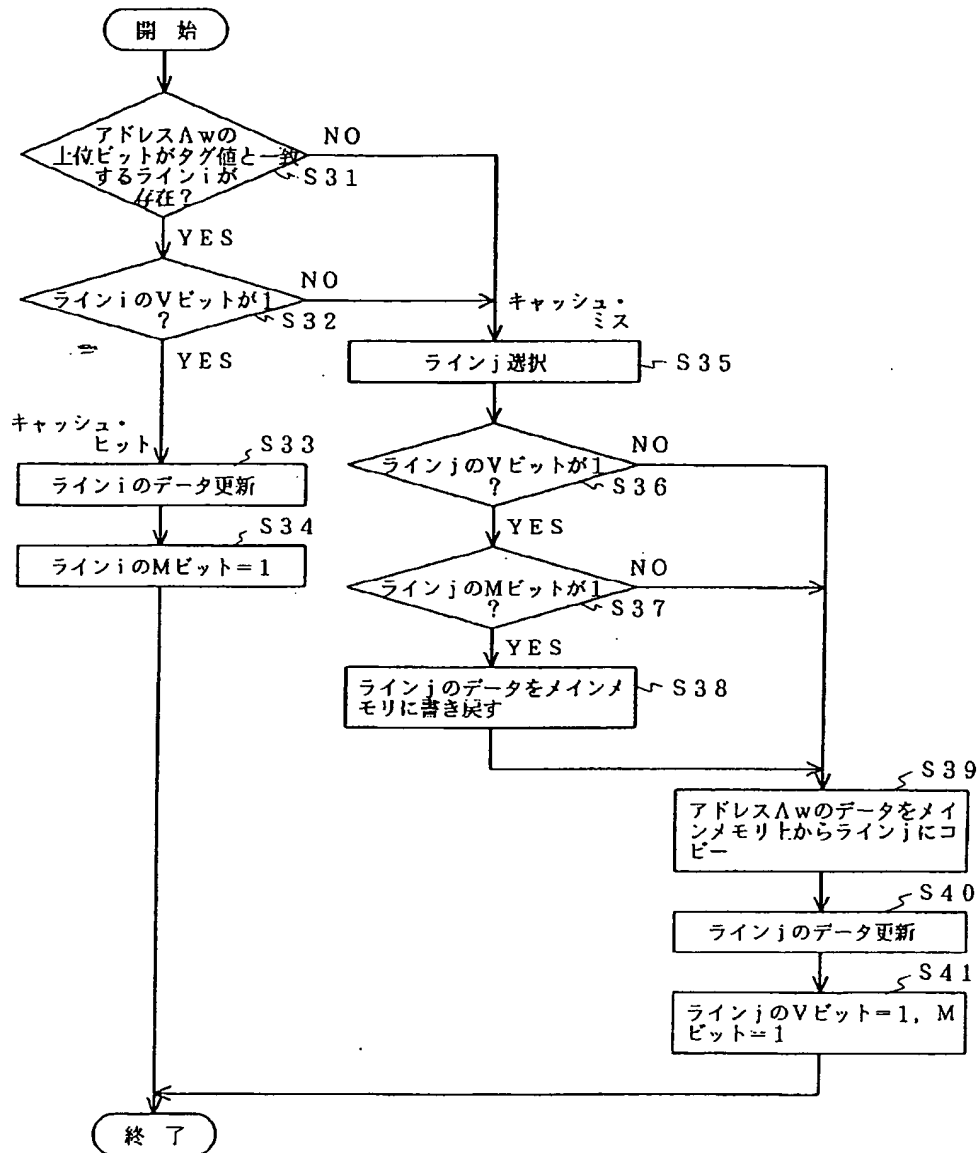
【図7】



【図5】



【図6】



[0060]

Figure 2 is a flow chart showing the operation during data writing according to the embodiment of the present invention. The operation during data writing according to the embodiment of the present invention will be described with reference to Figure 1 and Figure 2.

[0061]

Assume that a write request to write the address AW data to the cache memory 22 is provided from the general purpose processor. The memory control circuit 20 searches for a line 27 having a tag 23 value that matches the upper bit of the address Aw (Figure 2, step S1).

[0062]

If a line 27 having a tag 23 value matching the upper bit of the address Aw exists and this line is the line i, the memory control circuit 20 checks the V bit 24 of the line i (Figure 2, step S 2). If the V bit 24 is "1", the data of the line i is rewritten (Figure 2 the step S 3) because the line i includes the target data (cache hit). Also, the memory control circuit 20 sets the M bit of the line i to "1" (Figure 2, step S 4), to indicate that the data on the cache line i is newer than the data in the memory 21.

[0063]

In the cache memory 22, when the line having a tag 23 value that matches the upper bit of the address Aw does not exist or even when the line exists, the V bit 24 is "0", the target data is not present in the cache memory 22 (cache miss). Therefore, the memory control circuit 20 selects an appropriate line (the line j) (Figure 2, step S 5) and then writes the data to the line j (Figure 2, step S 9).

[0064]

In this regard, the memory control circuit 20 checks the V bit 24 and M bit 25 of the selected line j (Figure 2, step S 6 and S 7). If the both are "1", the line j holds data which is newer than the data in the main memory 21. Thus, the data of line j is written back to the corresponding address of the main memory 21 (Figure 2, step S 8), after which, data is written into the line j (Figure 2, step S 9).

[0065]

Subsequently, the memory control circuit 20 sets the V bit 24 and M bit 25 of the line j to "1" (Figure 2, step S 10), to indicate that the data on the cache line i is newer than the data in the main memory 21. In the same manner as in the case of a read cache miss, the value j of the line 27 is appropriately selected according to the address Aw value and the use condition of each line. As this method is publicly known, detailed description shall be omitted.

[0066]

Figure 3 shows the operation during data writing according to the embodiment of the present invention. How to speed up the process during a cache miss will be described based on the embodiment of the present invention with reference to Figure 3.

[0067]

As in the operation shown in Figure 7, Figure 3 shows the exchange of data between the main memory 21 and the cache memory 22 that is controlled by the cache memory control method, when data is written into consecutive addresses. It also shows the changing of data in the cache memory 22 and the main memory 21.

[0068]

In this case, as shown in Figure 7, it is assumed that the data size of a cache line is 16 bytes and the size of data to be written is 4 bytes, and subsequently, these four 4 byte data N0, N1, N2, and N3 are written one by one from the address Aw of the main memory 22. Also, as in Figure 7, the data of address Aw is not present in the cache memory 22 at the start of the process and the cache line i, which is scheduled to hold the address Aw data, stores indefinite values: U0, U1, U2, and U3 (Refer to 31 of Figure 3).

[0069]

First, the operation in the case where a request to write the value N0 into the address Aw is provided is considered (Refer to (1) of Figure 3). As the address Aw data is not present in the cache, a cache miss takes place. However, unlike the conventional technology, in the embodiment of the present invention, data M0, M1, M2 and M3 for one line from the address Aw of the main memory 21 is not read into the cache line i, and among the data in line i, only U0 which corresponds to the 4 byte data from the address Aw of the line i data is overwritten with N0 (Refer to (2) and 32 of Figure 3).

[0070]

Among the line i data, the data corresponding to the addresses A_w+4 , A_w+8 and A_w+12 are the indefinite U_1 , U_2 and U_3 , which do not reflect the values M_1 , M_2 and M_3 of the main memory 21. However, in the judgment of a cache hit/miss, line i is regarded to hold 16 bytes of data per line, from the address A_w of the main memory 21.

[0071]

Next, the 4 byte data N_1 is written to the adjacent address A_w+4 (Refer to (3) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address A_w is present in the cache memory 22, a hit occurs in the cache for this data N_1 write operation, and data corresponding to the address A_w+4 in the line i is updated with N_1 (Refer to 33 of Figure 3).

[0072]

Subsequently, the 4 byte data N_2 is written to the adjacent address A_w+8 (Refer to (4) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address A_w is present in the cache memory 22, a hit occurs in the cache for this data N_2 write operation, and data corresponding to the address A_w+8 in the line i is updated with N_2 (Refer to 34 of Figure 3).

[0073]

Finally, the 4 byte data N_3 is written into the adjacent address A_w+12 (Refer to (5) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address A_w is present in the cache memory 22, a hit occurs in the cache for this data N_3 write operation, and data corresponding to the address A_w+12 in the line i is updated with N_3 (Refer to 35 of Figure 3).

[0074]

As described above, all data on the cache line i is rewritten with N_0 , N_1 , N_2 and N_3 , and the data is newer than the 16-byte line data 36 from the address A_w in the main memory 21. Therefore, when a different address data is stored in the line i , the data in the line i should be written back to the main memory 21 [Refer to 37 and (6) of Figure 3].

[0075]

Comparing the operation from the embodiment of the present invention shown in Figure

3 with the conventional operation example shown in Figure 7, after the value N3 is written into the address Aw, the same result is obtained. In other words, the cache line i holds the values N0, N1, N2 and N3 (Refer to 35 of Figure 3 and 46 of Figure 7).

[0076]

However, whereas reading from the main memory 21 to the cache line i is executed in the conventional example ((17) of Figure 7) in order to obtain this result, such reading is not executed in the embodiment of the present invention. Thus, in the embodiment of the present invention, the result can be obtained faster than in the conventional example.

[0077]

Figure 3 describes the reason why reading of data from the main memory 21 to a cache line is not necessary at the time of a cache miss. In the case where all data in a cache line i is updated by a number of write operations, even when data from the main memory 21 is transferred to the cache line i due to a cache miss caused by the first write operation, all data transferred from the main memory is overwritten by plural write operations. Therefore, the final result stored in the line i is only dependent on the write data value and is not dependent on the content of the main memory 21. Thus, the reading of data from the main memory 21 to the cache line is not necessary.

[0078]

Generally, all data in a cache line is not guaranteed to be updated by plural write operations. Therefore, to maintain the consistency between the content of the main memory 21 and the content of the corresponding cache line data, it is necessary to read the content of the main memory 21 to the cache line when a cache miss occurs. However, in multimedia applications, all data on a cache line is guaranteed to be updated by plural write operations.

[0079]

For example, it is assumed that a compressed video image is decompressed by using software. Since the decompressed video image is successively written into an approximately several hundred-kilobyte frame buffer, all content of the cache line corresponding to the frame buffer area is updated. In multimedia applications, especially in applications for video images, a large amount of data is often written into the frame buffer, contributing to the acceleration of the process by omitting unnecessary read operations.

[0080]

A general write command (write, store) based on the conventional cache control method shown in Figure 6 and the write command based on the cache control method from the embodiment of the present invention shown in Figure 2 are prepared as separate commands in a command set 10 in the general purpose processor 1 and used in accordance with the intended use by an application software programmer.

[0081]

This is because it is impossible to judge if all data on a cache line can be updated by plural write operations in any hardware unit on the general purpose processor 1. In addition, it is difficult for a high level language compiler to make such a judgment.

[0082]

As just described, a general write command (write, store) 11, based on the conventional cache control method in the command set 10 of the general purpose processor 1 and the write command 12 for a cache miss, based on the cache control method in the embodiment of the present invention shown in Figure 2 are prepared as separate commands. When cache miss processing is required, the cache line is updated with the write data and the V bit and M bit of the line is set to "1" without reading data from the address of the main memory 21 corresponding to the write address according to the command 12 for a cache miss, thereby resulting to the updating of all data on the cache line. In this case, it is possible to omit the reading from the main memory 21 to the cache line, and speed up the process.

[0083]

This makes it possible to reduce the time for reading data from the address of the main memory 21. Thus, cache miss processing in writing a large amount of data to consecutive addresses can be speeded up and the execution speed of software including the process for a cache miss can be improved.

[0084]

[Effects of the Invention]

As described above, according to the present invention, in the main memory and the cache memory system including a cache memory for maintaining part of data stored in the main memory, a cache memory address is selected to store the write cache data at

the time of a cache miss. It is judged whether the cache memory data corresponding to the selected address is valid or invalid. If the data is judged as being invalid, data in the write cache is written into the selected address without reading the main memory data corresponding to the data to the cache memory. This makes it possible to speed up cache miss processing in writing a large amount of data to consecutive addresses and improve the execution speed of software including cache miss processing.

Figure 1

【図1】 General purpose processor

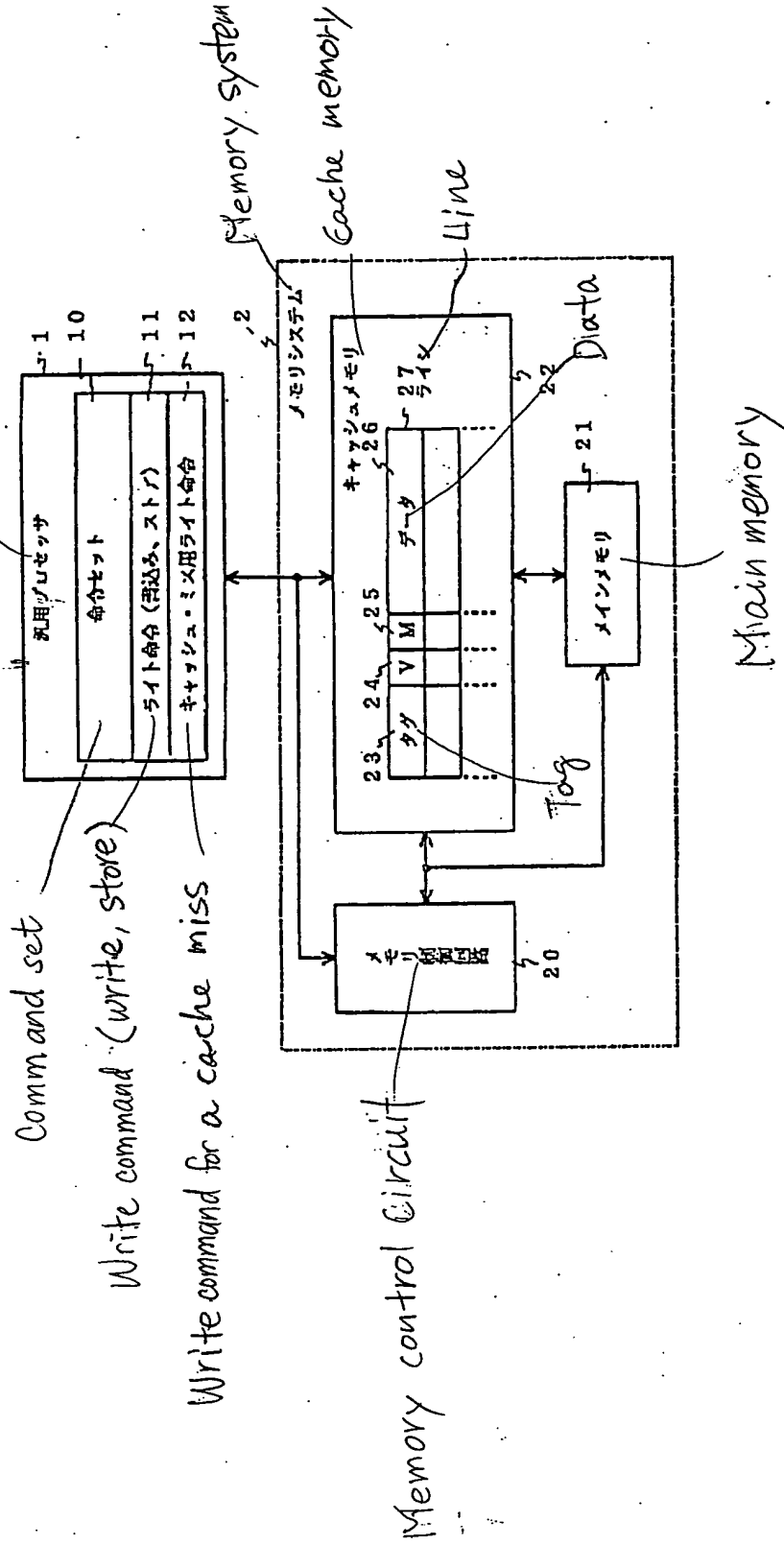


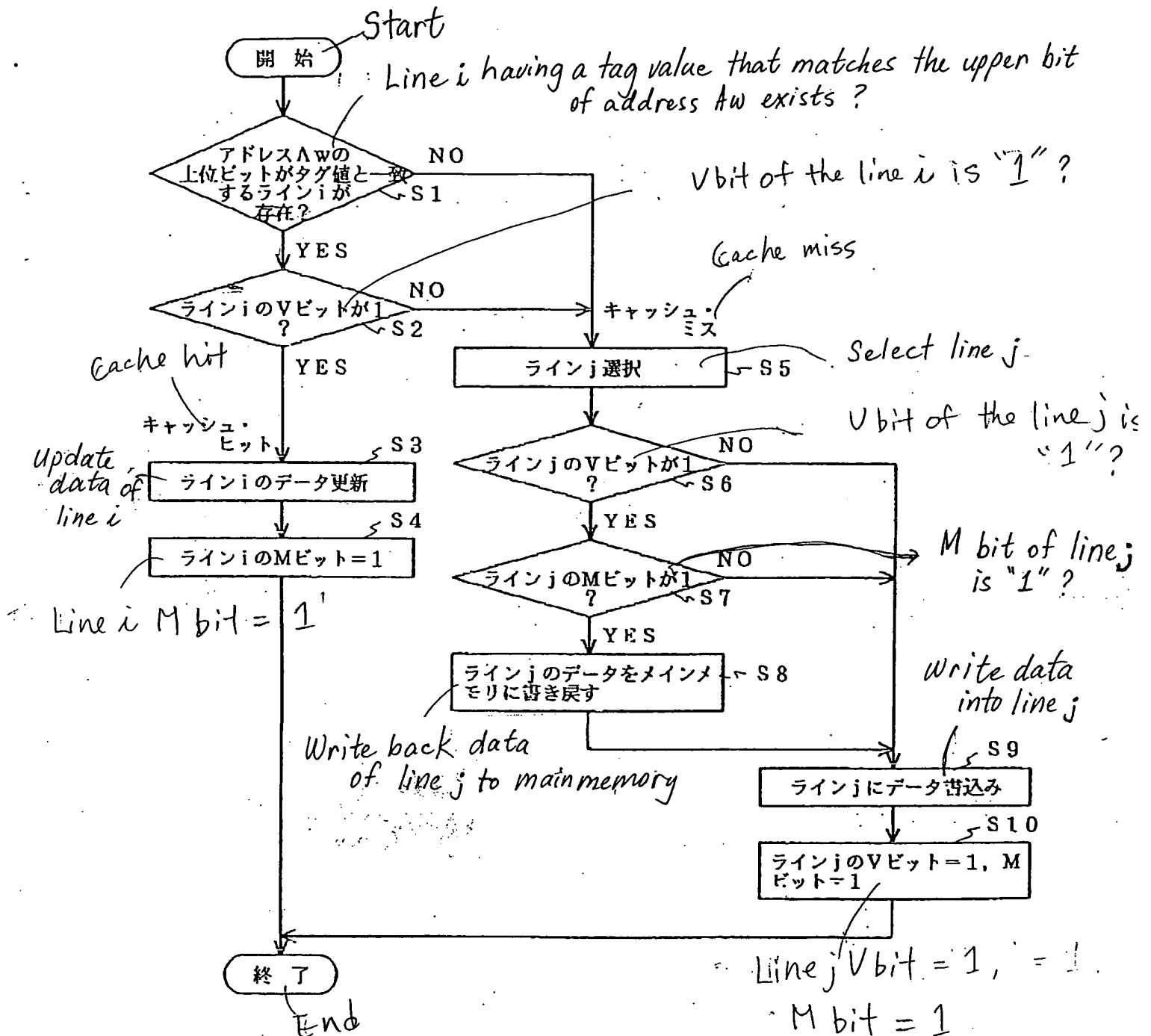
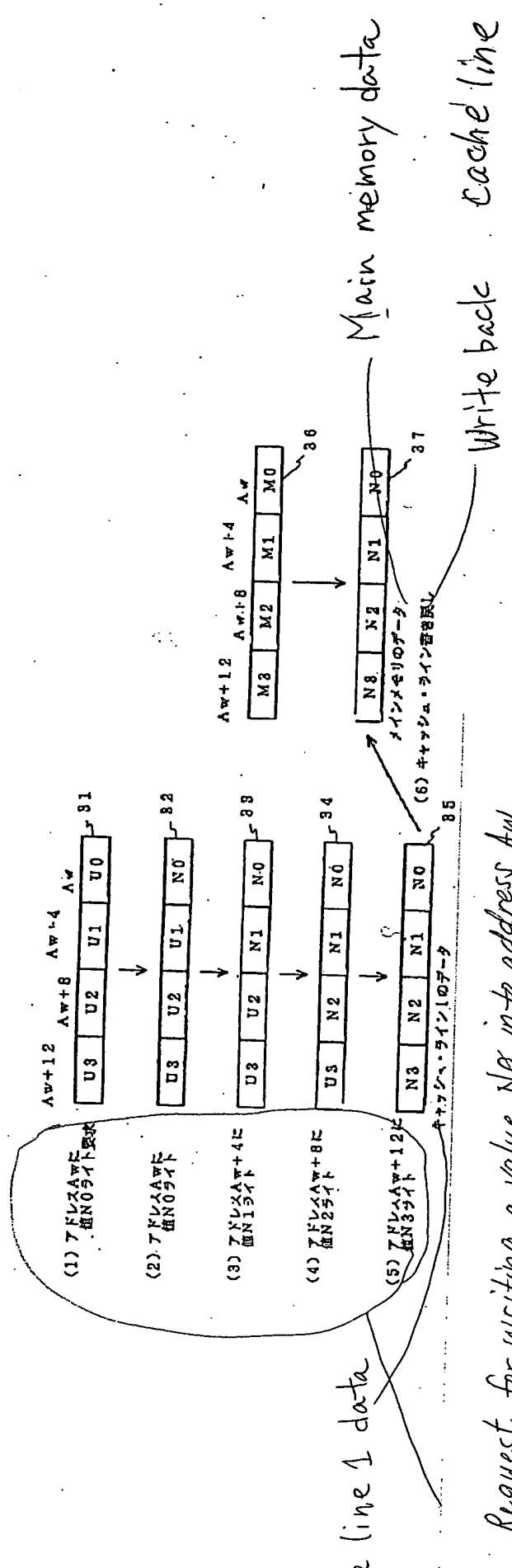
Figure 2
【図2】

Figure 3
【図3】



- Request for writing a value N0 into address Aw
- Write value N0 into address Aw
 - Write value N1 into address Aw+4
 - Write value N2 into address Aw+8
 - Write value N3 into address Aw+12

Figure 6

【図6】

Line i having a tag value that matches the upper bit of address Aw exists?

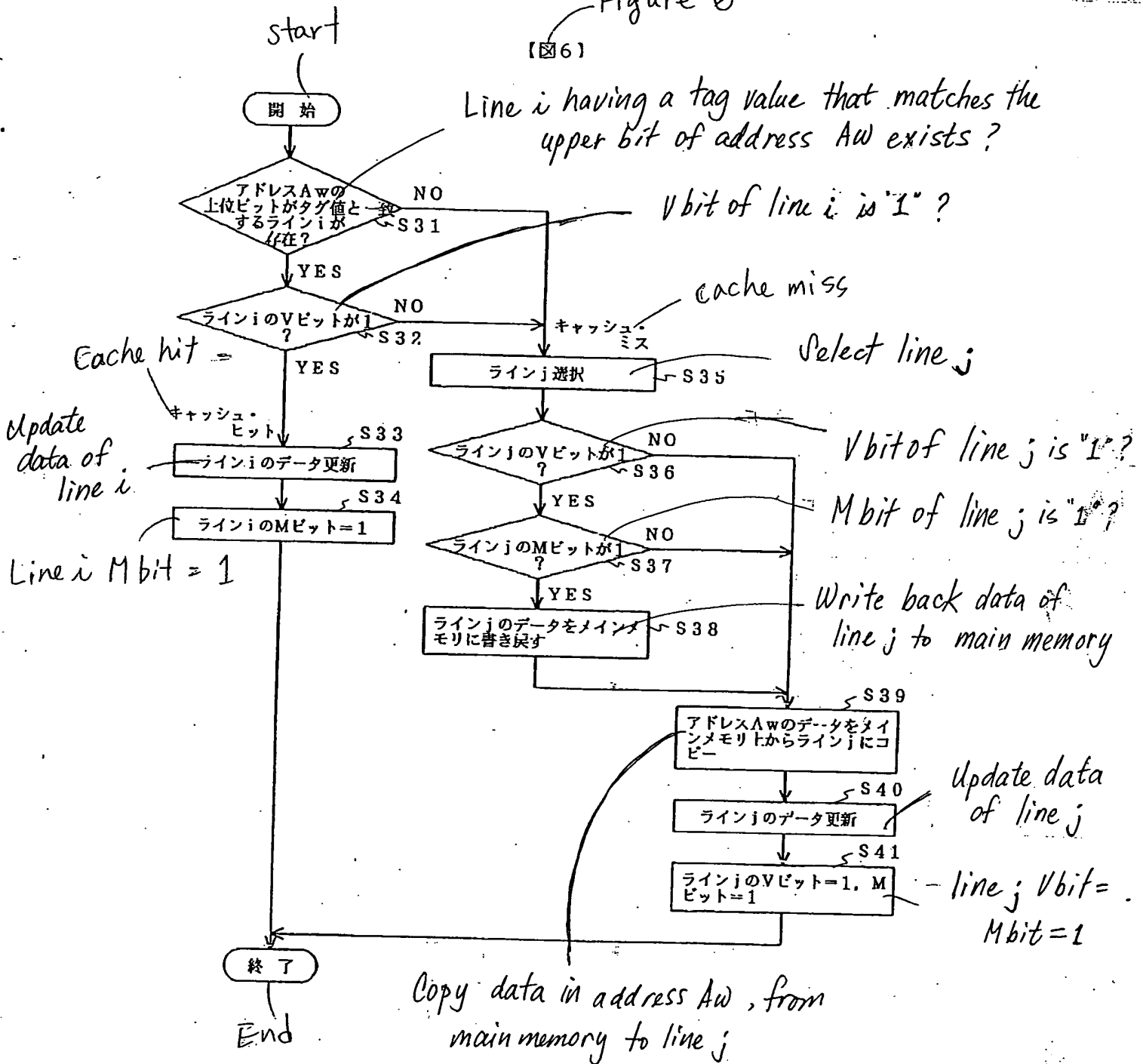
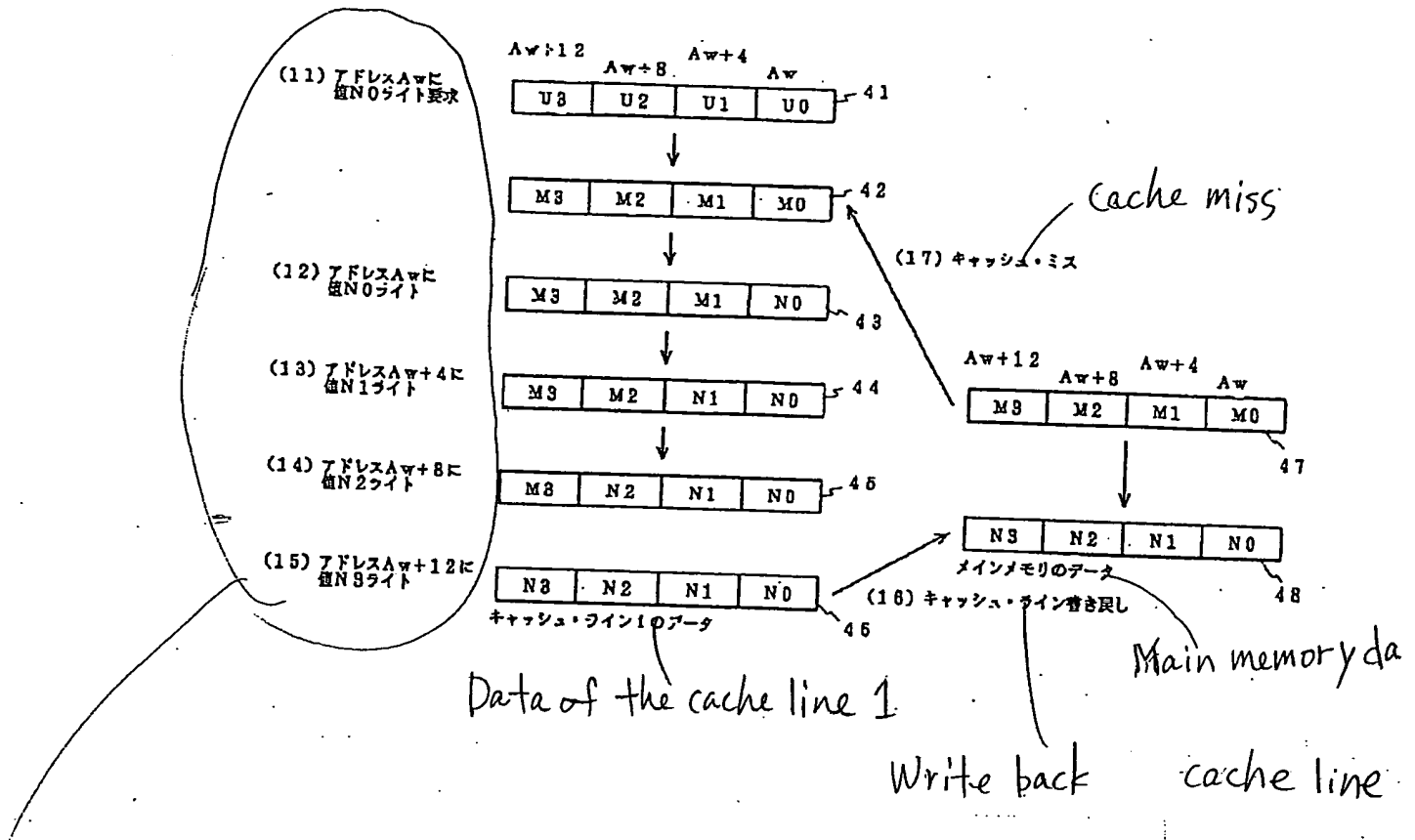


Figure 7
[7]



- (11) Request for writing value N0 into address Aw
- (12) Write value N0 into address Aw
- (13) Write value N1 into address Aw+4
- (14) Write value N2 into address Aw+8
- (15) Write value N3 into address Aw+12

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

☐ BLACK BORDERS

☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES

☐ FADED TEXT OR DRAWING

☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING

☐ SKEWED/SLANTED IMAGES

☒ COLOR OR BLACK AND WHITE PHOTOGRAPHS

☐ GRAY SCALE DOCUMENTS

☐ LINES OR MARKS ON ORIGINAL DOCUMENT

☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY

☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.